Automated muscle path calibration with gradient-specified optimization based on moment arm

Ziyu Chen[1,2], **Tingli Hu**[2,3], **Sami Haddadin**[2,3], **David W. Franklin**[1,2,4*]

***For correspondence:**
david.franklin@tum.de

[1]Neuromuscular Diagnostics, TUM School of Medicine and Health, Technical University of Munich, Munich, Germany; [2]Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Munich, Germany; [3]Chair of Robotics and Systems Intelligence (RSI), TUM School of Computation, Information and Technology, Technical University of Munich, Munich, Germany; [4]Munich Data Science Institute (MDSI), Technical University of Munich, Munich, Germany

Abstract Muscle path modeling is more than just routing a cable that visually represents the muscle, but rather it defines how moment arms vary with different joint configurations. The muscle moment arm is the factor that translates muscle force into joint moment, and this property has an impact on the accuracy of musculoskeletal simulation. However, it is not easy to calibrate muscle path based on moment arm, because each path is configured by various parameters while the relations between moment arm and both the parameters and joint configuration are complicated. Here, we tackle this challenge in the simple fashion of optimization, but with an emphasis on the gradient; when specified in its analytical form, optimization speed and accuracy will be improved. We explain in detail how to differentiate the enormous cost function and how our optimization is configured, then we demonstrate the performance of this method by fast and accurate replication of muscle paths from a state-of-the-art shoulder–arm model. As long as the muscle is represented as a cable wrapping around obstacles, our method could overcome difficulties in path calibration, both for developing generic models and for customizing subject-specific models.

Introduction

In musculoskeletal modeling, the geometry of a muscle is often represented by a series of straight and curved cables, namely the muscle path. It is defined by the locations of the origin, via, and insertion points as well as the size(s), location(s), and orientation(s) of the obstacle(s) around which the cables wrap (***Garner and Pandy, 2000***; ***Hicks et al., 2015***; ***Guo et al., 2020***; ***Lloyd et al., 2021***). Although such path is a geometrical simplification of the muscle, which alters properties related to the 3-D architecture, it can still have similar muscle length–joint angle and moment arm–joint angle relations to its anatomical reference, provided the path is well configured (***Delp, 1990***; ***Holzbaur et al., 2005***; ***Arnold et al., 2010***; ***Ma'touq et al., 2019***). These two relations are crucial to simulation accuracy since muscle length is a major variable in the contraction dynamics (***Gordon et al., 1966***; ***Gollapudi and Lin, 2009***; ***Millard et al., 2013***; ***Chen and Franklin, 2023***; ***Millard et al., 2024***) and moment arm directly determines the kinetic capacity of a muscle (***Weinstein et al., 1987***; ***Aper et al., 1996***; ***Sherman et al., 2013***).

The calibration of muscle path is not only a fundamental step for the development of a generic

musculoskeletal model, but also necessary in subject-specific applications (***Arnold et al., 2000***; ***Garner and Pandy, 2001***; ***Scheys et al., 2008***; ***Killen et al., 2021***; ***Ao et al., 2023***). Scaling of skeletal geometry, for example, is a common practice of model individualization, where the sizes and shapes of the subject's segments may be replicated in the model based on anthropometric measurements or inverse kinematics (***Delp et al., 2007***; ***Hicks et al., 2015***). However, such a scaled model is not yet subject-specific, because the individual characteristics of muscle length and moment arm are only reflected in musculoskeletal geometry, which is different from skeletal geometry (***Hicks et al., 2015***; ***Chen and Franklin, 2023***). In fact, the scaling of skeletal geometry might distort the biomechanical characteristics which were calibrated to be generically correct. For example, depending on how the path of the triceps surae is defined, enlarging the tibia might result in the Achilles tendon moment arm being increased, unchanged, or even decreased. Thus, for a subject-specific model, a rework of path calibration is necessary after scaling to reflect the individual characteristics of muscle length and moment arm, or at least assure that they remain similar to those in the generic model.

Muscle path calibration is performed based on experimental data such as geometric coordinates from medical images (***Arnold et al., 2000***; ***Garner and Pandy, 2001***; ***Scheys et al., 2008***; ***Modenese and Kohout, 2020***) or moment arm measurements (***Holzbaur et al., 2005***; ***McFarland et al., 2023***), and this process can be laborious. To begin with, there is a large number of path-related parameters to be tuned. Each muscle path is defined by at least six parameters, including the two 3-D coordinates of the origin and insertion points. In addition, there will be three extra parameters for each via point and many more when obstacles are included to recreate the anatomical feature of muscle geometry (***Delp, 1990***; ***Garner and Pandy, 2001***). For example, in the prevalent open-source modeling platform OpenSim (***Delp et al., 2007***), cylinders are often used as obstacles to prevent the muscle from penetrating into the joint (***Arnold et al., 2010***; ***Rajagopal et al., 2016***; ***McFarland et al., 2023***), where the cylinder size, orientation, and location would respectively need one, two, and three parameters to define. In general, monoarticular muscles require between six to 12 parameters, whereas complicated multiarticular muscles may require up to 30 parameters (***Garner and Pandy, 2001***).

The size of parameters is not necessarily a big challenge when one knows which to tune. However, if an objective function is too complex, the sensitivity to each parameter is often unclear (***Ao et al., 2023***). Hence, manual tuning tends to be puzzling: e.g., enlarging the obstacle or shifting it away from the center of rotation might both increase the corresponding moment arm, but the effect may differ when the joint configuration changes. Even with optimization, the tuning process is not labor-free, if weighting factors are involved and need to be tested repeatedly due to the absence of any knowledge regarding the sensitivity.

The second challenge in muscle path calibration lies in the high dimensional relation between moment arm and joint configuration. Each muscle, even monoarticular ones, can actuate multiple degrees of freedom (DoF), and each DoF corresponds to one moment arm (***Wretenberg et al., 1996***; ***McCullough et al., 2011***; ***Vaarbakken et al., 2014***). For instance, the gastrocnemius is a biarticular muscle crossing the knee and ankle, which can be considered to contain at least three DoFs (ankle plantar-/dorsiflexion, ankle eversion/inversion, knee extension/flexion). Therefore, besides the well-studied Achilles tendon moment arm in the sagittal plane (***Maganaris et al., 2000***; ***Ward et al., 2009***; ***Fath et al., 2010***; ***Holzer et al., 2020***), it has an extra ankle moment arm (***Hintermann et al., 1994***; ***Lee and Piazza, 2008***) and a moment arm about the knee (***Herzog and ter Keurs, 1988***; ***Wretenberg et al., 1996***). With all DoFs considered, path calibration involves matching multiple moment arm–joint angle relations altogether, and it is common to run into trouble where the calibration of one moment arm can distort others that are previously calibrated.

There is more to this challenge than only the high dimensional relation, because each moment arm is also affected by all actuating DoFs of the muscle (***Hintermann et al., 1994***; ***Delp et al., 1999***; ***Wolfram et al., 2018***). Suppose the knee moves while immobilizing the ankle, and in theory the Achilles tendon ankle moment arm might still change despite no ankle motion. With this, the mo-

ment arm–joint angle relation is no longer depictable by a curve or a surface, but rather requires a hypersurface to demonstrate. For example, the soleus has two moment arms around the ankle, and trying to calibrate them is similar to matching two pairs of surfaces (imagine two heatmaps), which is difficult but viable. However, for the gastrocnemius, each of its three moment arms is dependent on three DoFs, which means that depicting the relation of each moment arm with the DoFs is similar to plotting a volumetric heatmap in 3-D space. In this case, the task of calibration becomes matching the color for three pairs of 3-D heatmaps. While difficult to imagine, the manual calibration could still be theoretically possible if they are somehow matched slice by slice, where each slice is a 2-D heatmap. But with one more moment arm, e.g. for the rectus femoris or many of the shoulder muscles, the number of dimensions to be matched is beyond three, forcing manual tuning to be simplified and compromising the overall model accuracy.

In light of these challenges, it takes extensive effort to develop musculoskeletal models to simulate many different motions or to individualize them for each subject. This difficulty hinders the advancement and application of musculoskeletal modeling and simulation. Therefore, here we develop a gradient-based method for automated muscle path calibration. Our goal is to tune path-related parameters so that the moment arm–joint angle relation of a model matches with the target specification. An optimization framework is established for a classic muscle path wrapping method (***Garner and Pandy, 2000***) and the gradient for the cost function is derived in its analytical form to increase optimization speed and accuracy. The concept employed in the gradient derivation is universal and may be applied to the model calibration of many other complex systems.

Methods

Optimization Method

The process of parameter tuning for a muscle path is formulated as a least-squares problem with the following cost function

$$J(\boldsymbol{p}) = \sum_{n=1}^{N} \left\| \boldsymbol{r}_{\text{target}}(\boldsymbol{q}_n) - \boldsymbol{r}_{\text{model}}(\boldsymbol{q}_n, \boldsymbol{p}) \right\|_2^2 \rightarrow \min, \tag{1}$$

where $\boldsymbol{p}$ denotes muscle path parameters, and $\boldsymbol{r}(\boldsymbol{q})$ is the moment arm at joint configuration $\boldsymbol{q}$. Note that $\boldsymbol{r}$ and $\boldsymbol{q}$ are vectors with the dimension of DoF number, whereas N is the number of joint configurations. The target value and the model output are indicated by the subscripts, but to make the equations concise, the model output will also be abbreviated as $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$, $\boldsymbol{r}(\bullet, \boldsymbol{p})$, or $\boldsymbol{r}$.

In this study, we limited the obstacle type to cylinder to simplify the discussion, but the same principle applies to other types such as cube and stub described in ***Garner and Pandy*** (***2000***). Currently, our optimization method requires to decide in advance the number of cylinders for each muscle, and the composition of $\boldsymbol{p}$ varies across muscles with three typical forms:

1. No obstacle: $({}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}})$
2. Single cylinder: $({}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}, R, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta)$
3. Double cylinders: $({}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}, R_1, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}_1}, \alpha_1, \beta_1, R_2, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}_2}, \alpha_2, \beta_2)$

where ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}} \in \mathbb{R}^3$ are the coordinates of the origin and insertion points expressed in their respective joint frames, written as column vectors; e.g., ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} = [{}^{\mathrm{j}}x_{\mathrm{P}} \ {}^{\mathrm{j}}y_{\mathrm{P}} \ {}^{\mathrm{j}}z_{\mathrm{P}}]^{\mathrm{T}}$. The radius of a cylinder is denoted by $R \in \mathbb{R}^+$ and the center by ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}} \in \mathbb{R}^3$, whereas the orientation is defined by two Euler angles $\alpha, \beta \in \mathbb{R}$. In the following discussion, the origin and insertion points are denoted as P and S, and when wrapped by the cylinder (whose center is denoted as C), the two wrapping points are denoted as Q and T (***Figure 1***). Also, the frame in which coordinates are expressed are indicated by the superscript: with j for the joint frame, c for the cylinder frame, and w for the world frame.

Muscle path is configured by $\boldsymbol{p}$ using the obstacle-set method (***Garner and Pandy, 2000***). To reduce computational load, we first took a geometric approach to compute $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$, which is based on the principle of virtual work and algorithmically less complex by computing partial velocity terms

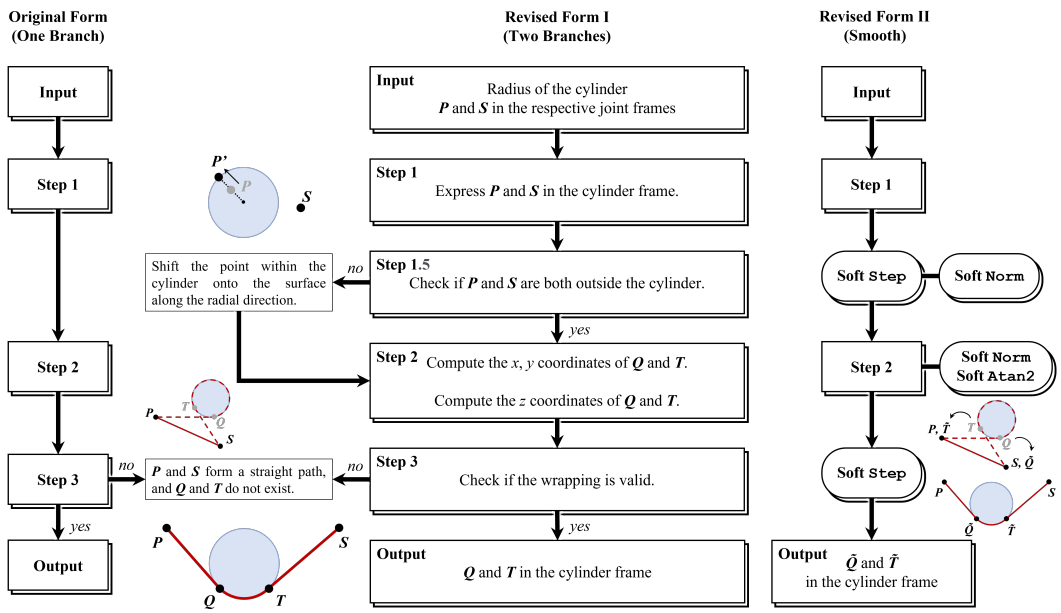


Figure 1. Workflows of the original and revised obstacle-set methods. Left: the original form with one conditional statement. Middle: the revised form with an extra conditional statement for continuity. Right: the smooth form revised for optimization. The naming and order of the origin, wrapping, and insertion points are indicated by the illustration in the bottom.

using the Kane's method (***Kane and Levinson, 1985***; ***Yamaguchi, 2001***; ***Sherman et al., 2013***). Then, since the typical algorithms solving ***Equation 1*** (e.g., Levenberg-Marquardt or trust-region) require the gradient $\partial J/\partial \boldsymbol{p}$, and by the chain rule

$$\frac{\partial J}{\partial \boldsymbol{p}} = \frac{\partial J}{\partial \boldsymbol{r}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \sum_{n=1}^{N}\left(2\left(\boldsymbol{r}(\boldsymbol{q}_n,\boldsymbol{p}) - \boldsymbol{r}_{\text{target}}(\boldsymbol{q}_n)\right)\right)\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}}, \tag{2}$$

we need to specify the gradient of $\boldsymbol{r}(\bullet,\boldsymbol{p})$ in its analytical form. For this, we began with revising the obstacle-set method into a smooth form, such that $\partial \boldsymbol{r}/\partial \boldsymbol{p}$ exists for all $\boldsymbol{p}$. This is accomplished by replacing the conditional statements in the obstacle-set method as well as other nondifferentiable components with soft functions which yield almost the same outputs but are continuously differentiable (***Figure 1***).

Direct derivation of the gradient is overwhelming considering the complexity of $\boldsymbol{r}(\bullet,\boldsymbol{p})$. It would require a tremendous amount of effort, and the eventual result would be filled with pages of repeated terms and computationally inefficient. Thus, we circumvented this problem by disassembling it into a composite of multiple gradients whose analytical forms are easy to derive.

Soft Functions

Figure 1 (middle) shows the detailed workflow of the revised obstacle-set method, containing two conditional statements. The first statement checks if either of the designated fixed points is within the cylinder. Note that this is not included in the original method by ***Garner and Pandy*** (***2000***) (***Figure 1***, left), and we introduce it to simplify the optimization since it removes complex geometric constraints between the wrapping object and the points. The second statement checks if the wrapping angle exceeds 180°, which is considered invalid wrapping. Naturally, conditional statements make $\boldsymbol{r}(\bullet,\boldsymbol{p})$ nondifferentiable in certain domains, and we introduced soft functions to patch them (***Figure 1***, right).

Step 1.5 essentially keeps a minimum distance of R between the point and the cylindrical axis (e.g., $\|{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}\|_2 = \sqrt{{}^{\mathrm{c}}x_{\mathrm{P}}{}^2 + {}^{\mathrm{c}}y_{\mathrm{P}}{}^2}$) to prevent Step 2 from returning `Inf` or `NaN`. In our case, for example if

the origin point P locates inside the cylinder, it is shifted onto the surface along the radial direction as P′ (***Figure 1***, middle), and the coordinates become

$$ {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}} = \frac{R}{\| {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy} \|_2} {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}. \tag{3} $$

More specifically,

$$ {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} = \begin{cases} {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}, & \| {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy} \|_2 > R \\ {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}}, & \| {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy} \|_2 \leq R \end{cases}, \tag{4} $$

and the coordinates of such transitional point $\tilde{\mathrm{P}}$ may be calculated using a soft version of Heaviside step function:

$$ {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} = {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}} + \left({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},\mathrm{xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}} \right) f_{\mathsf{softStep}}(d_{\mathrm{P}}), \tag{5} $$

where $d_{\mathrm{P}} = {}^{\mathrm{c}}x_{\mathrm{P}}{}^2 + {}^{\mathrm{c}}y_{\mathrm{P}}{}^2 - R^2$, and

$$ f_{\mathsf{softStep}}(x) = \frac{1}{e^{-2x} + 1} \approx \begin{cases} 1, & x > 0 \\ 0, & x < 0 \end{cases}; \tag{6} $$

the input can be magnified to steepen the transition around 0. Now, the conditional statement of ***Equation 4*** is smoothed: $\tilde{\mathrm{P}}$ remain as P if $d_{\mathrm{P}} > 0$, otherwise it approaches the shifted point P′ when d_{P} tends to 0 in the negative direction.

In Step 3, based on the sign of

$$ s = R\left({}^{\mathrm{c}}x_{\mathrm{Q}}{}^{\mathrm{c}}y_{\mathrm{T}} - {}^{\mathrm{c}}x_{\mathrm{T}}{}^{\mathrm{c}}x_{\mathrm{Q}} \right), \tag{7} $$

the method returns either the coordinates of the two wrapping points or null if wrapping is invalid (***Figure 1***, left). The latter situation can be problematic for optimization as it leads to yet another conditional statement in moment arm computation, in which different formulas are used depending on whether the wrapping occurs:

$$ \text{straight path: } \boldsymbol{r} = \left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{q}} \right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\| {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}} \|_2}, \tag{8} $$

$$ \begin{aligned} \text{wrapped path: } \boldsymbol{r} = & \left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}}}{\partial \boldsymbol{q}} \right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\| {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}} \|_2} \\ & + \left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{q}} \right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}}{\| {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}} \|_2}. \end{aligned} \tag{9} $$

Therefore, the revision of Step 3 into a smooth form involves a transition from ***Equation 8*** to ***Equation 9*** when s changes from positive to negative. Here, ${}^{\mathrm{w}}\boldsymbol{u}$ is transformed from ${}^{\mathrm{j}}\boldsymbol{u}$ or ${}^{\mathrm{c}}\boldsymbol{u}$:

$$ \begin{bmatrix} {}^{\mathrm{w}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}(\boldsymbol{q}) \begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}(\boldsymbol{q}) {}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}} \begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u} \\ 1 \end{bmatrix}, \tag{10} $$

where ${}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}(\boldsymbol{q})$ is the transformation matrix from the joint frame to the world frame, and ${}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}}$ cylinder to joint. The complication arises with involvement of the joint frames, indicating the bones on which P, S, and the cylinder are fixed: ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}$ and ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}$ must be defined in separate bones, one or both of which are different from the bone for fixating the cylinder. In other words, the implication of j in ***Equation 10*** is different between P and Q as well as between T and S, making it hard to cancel out the terms in ***Equation 9*** to transition into ***Equation 8***. To this end, we have Q and T respectively approaching S and P when s tends to 0 in the negative direction (***Figure 1***, right), and similarly this can be achieved with ***Equation 6***:

$$ \begin{aligned} {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} &= {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}} + \left({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}} \right) f_{\mathsf{softStep}}(s), \\ {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} &= {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} + \left({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} \right) f_{\mathsf{softStep}}(s). \end{aligned} \tag{11} $$

This way, moment arm computation is generalized as

$$\begin{aligned}\boldsymbol{r}(\boldsymbol{q}, {}^{\mathrm{w}}\boldsymbol{u}) = &\left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\|\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}\|_2} \\ &+\left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\|\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}\|_2}.\end{aligned} \tag{12}$$

When $s > 0$, $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ maintain their original coordinates as wrapping points Q and T (${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}$ and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}$), and moment arm is computed with ***Equation 9***. When $s < 0$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}$ and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}$, and ***Equation 12*** simplifies into ***Equation 8***. Importantly, this generalization accommodates the wrapping and moment arm computation involving two obstacles.

Furthermore, Step 2 involves calculation of the 2-norm (also in ***Equations 3*** and ***12***) and the 2-argument arctangent. Both are nondifferentiable at the origin, and we replaced them with

$$f_{\mathsf{softNorm}}(\boldsymbol{x}) = \sqrt{\|\boldsymbol{x}\|_2{}^2 + a} - \sqrt{a}, \tag{13}$$

where a should be a sufficiently small positive number (e.g., $\sqrt{\mathtt{eps}}$), and

$$f_{\mathsf{softAtan2}}(x, y) = 2\arctan\left(\frac{y}{\sqrt{x^2 + y^2} + x}\right), \tag{14}$$

where x is assigned a negative value with a sufficiently small magnitude (e.g., -0.001), if x is negative but greater than this value. This prevents the denominator term in ***Equation 14*** from becoming 0 and hence avoid discontinuity.

In such a manner, we have modified a branched workflow into a smooth form that is differentiable across the entire domain of its input parameters. This modification is verified using kinematic measurements to ensure that its difference with the non-smooth form is small enough across a large range of motion (***Appendix 1***).

Gradients

With ***Equation 6***, the obstacle-set method, or more precisely, the computation of $\boldsymbol{u}(\boldsymbol{q}, \boldsymbol{p})$ is revised into a smooth form, and since ***Equations 6***, ***13*** and ***14*** are clearly all differentiable, it is now possible to compute the gradient $\partial\boldsymbol{u}/\partial\boldsymbol{p}$.

Our target function $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$ is a composite of $\boldsymbol{u}(\boldsymbol{q}, \boldsymbol{p})$ and the generalized moment arm computation $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{u})$ with ***Equation 12***, and by the chain rule, we may now have

$$\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}}, \tag{15}$$

which simplifies the derivation by disassembling a complex function into the product of less complicated ones. However at this stage, ***Equation 15*** is still difficult to compute due to the knotty structure of $\partial\boldsymbol{u}/\partial\boldsymbol{p}$, so we continue down the chain.

Note that in this study, the gradient of a scalar variable is formatted as a row vector, while that of a vector variable is formatted as a matrix, namely the Jacobian, whose rows are the gradients of each scalar variable. Suppose for the single-cylinder case with 12 parameters, $\partial R/\partial\boldsymbol{p}$ is simply a 12-D row vector whose seventh element is 1 while the rest are 0, and $\partial\, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}$ is a 3×3 identity matrix concatenated with a 3×9 null matrix.

In Step 1, the coordinates of the origin and insertion points are transformed from the joint frame to the cylinder frame

$$\begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta) \begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u} \\ 1 \end{bmatrix}, \tag{16}$$

and ${}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta)$ is a 4×4 transformation matrix, from the joint frame to the cylinder frame, determined by the location and orientation of the cylinder

$$
{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}} = {}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}}{}^{-1} = \left[\begin{array}{ccc:c} \cos\beta & 0 & \sin\beta & {}^{\mathrm{j}}x_{\mathrm{C}} \\ \sin\alpha\sin\beta & \cos\alpha & -\cos\beta\sin\alpha & {}^{\mathrm{j}}y_{\mathrm{C}} \\ -\cos\alpha\sin\beta & \sin\alpha & \cos\alpha\cos\beta & {}^{\mathrm{j}}z_{\mathrm{C}} \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right]^{-1}. \tag{17}
$$

Based on the product rule, we may then easily compute the gradient (with the origin point as an example)

$$
\begin{aligned}
\underbrace{\begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} \\ 0 \ldots 0 \end{bmatrix}}_{4\times 12} &= \left[\begin{array}{c:c:c:c} \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}x_{\mathrm{P}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \cdots \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}z_{\mathrm{S}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 6} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial R}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 1} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}x_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \cdots}_{4\times 3} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial \alpha}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \cdots}_{4\times 2} \end{array}\right] \\
&\qquad + {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}} \underbrace{\begin{bmatrix} \frac{\partial\, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} \\ 0 \ldots 0 \end{bmatrix}}_{4\times 12} \\
&= \left[\begin{array}{c:c:c:c} \underset{4\times 6}{\mathbf{O}} & \underset{4\times 1}{\mathbf{O}} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}x_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \quad \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}y_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \quad \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}z_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 3} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial \alpha}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \quad \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial \beta}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 2} \end{array}\right] \\
&\qquad + {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}} \begin{bmatrix} \underset{3\times 3}{\mathbf{I}} & \underset{3\times 9}{\mathbf{O}} \\ 0 \ \ldots & 0 \end{bmatrix},
\end{aligned} \tag{18}
$$

and notice the partial derivatives with respect to $\boldsymbol{p}$ are in the order of ${}^{\mathrm{j}}x_{\mathrm{P}}$, ${}^{\mathrm{j}}y_{\mathrm{P}}$, ${}^{\mathrm{j}}z_{\mathrm{P}}$, ${}^{\mathrm{j}}x_{\mathrm{S}}$, ${}^{\mathrm{j}}y_{\mathrm{S}}$, ${}^{\mathrm{j}}z_{\mathrm{S}}$, R, ${}^{\mathrm{j}}x_{\mathrm{C}}$, ${}^{\mathrm{j}}y_{\mathrm{C}}$, ${}^{\mathrm{j}}z_{\mathrm{C}}$, α, and β.

Step 1.5 checks if the origin and insertion points P and S are both outside the cylinder, and we utilized ***Equation 6*** to return adjusted coordinates as well as their gradients (with the x coordinate of the origin point as an example)

$$
{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}} = {}^{\mathrm{c}}x_{\mathrm{P}'} + \left({}^{\mathrm{c}}x_{\mathrm{P}} - {}^{\mathrm{c}}x_{\mathrm{P}'}\right) f_{\mathsf{softStep}}(\delta_{\mathrm{P}}), \tag{19}
$$

$$
\frac{\partial\, {}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} = \frac{{}^{\mathrm{c}}x_{\mathrm{P}'}}{\partial \boldsymbol{p}} + \frac{\partial\left({}^{\mathrm{c}}x_{\mathrm{P}} - {}^{\mathrm{c}}x_{\mathrm{P}'}\right)}{\partial \boldsymbol{p}} f_{\mathsf{softStep}}(\delta_{\mathrm{P}}) + \left({}^{\mathrm{c}}x_{\mathrm{P}} - {}^{\mathrm{c}}x_{\mathrm{P}'}\right) \frac{\partial f_{\mathsf{softStep}}(\delta_{\mathrm{P}})}{\partial \boldsymbol{p}}, \tag{20}
$$

where all of the partial derivative terms can be easily derived using the chain rule. For instance,

$$
\begin{aligned}
\frac{\partial f_{\mathsf{softStep}}(\delta_{\mathrm{P}})}{\partial \boldsymbol{p}} &= \frac{\partial f_{\mathsf{softStep}}}{\partial \delta_{\mathrm{P}}} \left[\begin{array}{c:c} \underbrace{\frac{\partial \delta_{\mathrm{P}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}}_{1\times 3} & \frac{\partial \delta_{\mathrm{P}}}{\partial R} \end{array}\right] \underbrace{\begin{bmatrix} \partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} / \partial \boldsymbol{p} \\ \hdashline \partial R / \partial \boldsymbol{p} \end{bmatrix}}_{4\times 12} \\
&= \frac{2}{(e^{-2\delta_{\mathrm{P}}}+1)(e^{2\delta_{\mathrm{P}}}+1)} \; 2 \left[\begin{array}{ccc:c} {}^{\mathrm{c}}x_{\mathrm{P}} & {}^{\mathrm{c}}y_{\mathrm{P}} & 0 & -R \end{array}\right] \begin{bmatrix} \partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} / \partial \boldsymbol{p} \\ \hdashline \underset{1\times 6}{\mathbf{O}} \ 1 \ \underset{1\times 5}{\mathbf{O}} \end{bmatrix},
\end{aligned} \tag{21}
$$

where $\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} / \partial \boldsymbol{p}$ is the only variable unspecified in $\boldsymbol{p}$ and may be obtained from ***Equation 18***.

Next, with ${}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}$, ${}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}$, ${}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}$, and ${}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}$ from ***Equation 19***, we can derive the x and y coordinates of the wrapping points Q and T as described in ***Garner and Pandy*** (***2000***):

$$
\begin{aligned}
\begin{bmatrix} {}^{\mathrm{c}}x_{\mathrm{Q}} \\ {}^{\mathrm{c}}y_{\mathrm{Q}} \end{bmatrix} &= \begin{bmatrix} \frac{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}} R^2 - {}^{\mathrm{c}}y_{\tilde{\mathrm{P}}} R \sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2 - R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2} \\ \frac{{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}} R^2 + {}^{\mathrm{c}}x_{\tilde{\mathrm{P}}} R \sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2 - R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2} \end{bmatrix}, \\
\begin{bmatrix} {}^{\mathrm{c}}x_{\mathrm{T}} \\ {}^{\mathrm{c}}y_{\mathrm{T}} \end{bmatrix} &= \begin{bmatrix} \frac{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}} R^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{S}}} R \sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2 - R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2} \\ \frac{{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}} R^2 - {}^{\mathrm{c}}x_{\tilde{\mathrm{S}}} R \sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2 - R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2} \end{bmatrix},
\end{aligned} \tag{22}
$$

and additionally with the respective partial derivatives from ***Equation 20***, we can also compute their gradients

$$
\begin{aligned}
\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial R} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{P}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{P}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial R} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial R} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial R} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}},
\end{aligned}
\tag{23}
$$

where the first terms on the right side are easy to derive from ***Equation 22*** (so we omit the process to avoid expanding the results) and the second terms are obtained from ***Equations 18*** and ***20***.

Step 2 ends with the calculation of ${}^{\mathrm{c}}z_{\mathrm{Q}}$ and ${}^{\mathrm{c}}z_{\mathrm{T}}$, and Step 3 returns the coordinates of $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ after checking the validity of the wrapping (***Equation 11***). Likewise, their gradients are formatted as

$$
\begin{aligned}
\frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}},
\end{aligned}
\tag{24}
$$

$$
\begin{aligned}
\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}.
\end{aligned}
\tag{25}
$$

Here at the end of the obstacle-set method, we have obtained ${}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}(\boldsymbol{p})$, ${}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}(\boldsymbol{p})$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}(\boldsymbol{p})$, and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}(\boldsymbol{p})$ as well as their gradients. Before computing moment arm with ***Equation 12***, we need to utilize ***Equation 10*** to transform them from their respective joint frames to the world frame.

Finally, ***Equation 15*** may now be extended into

$$
\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \begin{bmatrix} \frac{\partial \boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial \boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial \boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}} & \frac{\partial \boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial \boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \tag{26}
$$

where the first term on the right side is easily derived from ***Equation 12***, while the second term lies in the chain of ***Equations 18***, ***20*** and ***23–25***. Our derivation of this compound gradient is verified as described in the next section.

Calibration and Validation

With the gradient $\partial \boldsymbol{r}/\partial \boldsymbol{p}$ specified, we used the nonlinear least-squares solver (`lsqnonlin`) in MATLAB to solve ***Equation 1***. The computation of this cost function requires a musculoskeletal model and the input of some kinematic dataset $\boldsymbol{q}$ for the model output, and the correspondent moment arm data at $\boldsymbol{q}$ are required as the target value. Ideally, $\boldsymbol{q}$ should cover joint configurations as diverse as possible for accurate calibration.

For the model output, we used a 12-DoF 42-muscle human shoulder–arm model (***Hu et al., 2020***) (with muscle path from ***Garner and Pandy*** (***2001***); see all DoFs and muscle paths in ***Appendix 2***), and the kinematic measurements of the shoulder and arm from 10 intransitive daily tasks performed by a single subject in ***Hu et al.*** (***2018***). The kinematics from five tasks (gesturing an OK sign, pumping fists, blocking out light from the face, saluting, and pointing) were input to the reference model to generate artificial calibration data, while $\boldsymbol{q}$ from another five tasks (gesturing a thumb-down, signaling for hitchhike, greeting, gesturing to stop, and gesturing for silence) were used for validation.

For the target value, we utilized the same model as reference to generate artificial moment arm data; that is, ***Equation 1*** becomes

$$
J(\boldsymbol{p}) = \sum_{n=1}^{N} \left\| \boldsymbol{r}_{\mathrm{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\mathrm{ref}}) - \boldsymbol{r}_{\mathrm{modelGP+}}(\boldsymbol{q}_n, \boldsymbol{p}) \right\|_2^2 \to \min, \tag{27}
$$

where $\boldsymbol{r}_{\text{modelGP}}$ is computed based on the slightly revised obstacle-set method by ***Garner and Pandy*** (***2000***) (***Figure 1***, middle) with $\boldsymbol{p}_{\text{ref}}$ from ***Garner and Pandy*** (***2001***), while $\boldsymbol{r}_{\text{modelGP+}}$ is based on our smoothed form (***Figure 1***, right). Thus in a sense, this intended calibration is equivalent to replicating the geometry of all muscle paths in the reference model. Due to the sample size of each kinematic measurement, we interpolated the original data to avoid redundant computation. For calibration, to avoid an underdetermined system, the number of joint configurations equals to the number of path parameters that each muscle has (***Appendix 2–table 2***); e.g., an 18-parameter path will be calibrated based on the kinematics in 18 instants from five movements (N as 18 in ***Equation 27***). For validation, five instants were extracted from each movement, that is 25 joint configurations in total.

Before the implementation, to determine an appropriate configuration for optimization and to evaluate the contribution of gradient specification, we tested two `lsqnonlin` algorithms using both analytical and numerical gradients. First, we tested the `levenberg-marquardt` algorithm on two single cylinder–based (serratus anterior, superior and middle parts) and two double cylinder–based muscle paths (latissimus dorsi, thoracic and iliac parts), each calibrated with 70 sets of initial points. The `levenberg-marquardt` algorithm is relatively simple in that it has only one type of search direction (***Nocedal and Wright, 2006***; ***The MathWorks Inc., 2024***), and the optimization process should generally be similar when the analytical and numerical gradients differ only by numerical error. So with this test, we may also verify if our computation of the analytical gradient is correct. The initial points for each path were randomly generated using `rand` in five ranges (from $[-1, 1]$ to $[-1e4, 1e4]$ with an increment of magnitude in between) and optimized without any constraints for our method to be verified across a vast domain. Then, we tested `trust-region-reflective`, which is the default algorithm for `lsqnonlin` and more robust. Calibration was performed with two additional complex paths (extensors carpi radialis brevis and ulnaris), each with 84 sets of initial points. Unlike in the first test, these initial points were randomly generated and bounded with the anatomy slightly taken into account, as will be described later. For both tests, `FunctionTolerance`, `OptimalityTolerance`, and `StepTolerance` were configured as 2.5e-16 with `MaxIterations` as 1e3 and `MaxFunctionEvaluations` as 1e5, and the numerical gradient was computed using MATLAB's built-in forward-difference method.

Based on the preliminary results, our intended calibration of 42 muscle paths is configured as follows:

- `trust-region-reflective` (gradient-specified);
- 1e-4 for `StepTolerance`, and 2.5e-16 for `FunctionTolerance` and `OptimalityTolerance`;
- 1e2 for `MaxIterations` and `Inf` for `MaxFunctionEvaluations`,

and the initial points were generated with the subsequent considerations.

As previously mentioned, each muscle path requires predetermination of the number of via points and cylinders for the structure of $\boldsymbol{p}$. Some muscle paths are modeled with via points and more than two obstacles (***Appendix 2–table 2***), and their structures of $\boldsymbol{p}$ will be a combination of the three typical forms. For example, the extensor carpi radialis longus is modeled in the form of

$$\underbrace{\text{origin—via}}_{\boldsymbol{p}_{\text{straight}}}\overbrace{\text{—cylinder—via}}^{\boldsymbol{p}_{\text{single}}}\underbrace{\text{—via}}_{\boldsymbol{p}_{\text{straight}}}\overbrace{\text{—cylinder—via}}^{\boldsymbol{p}_{\text{single}}}\underbrace{\text{—insertion}}_{\boldsymbol{p}_{\text{straight}}},$$

which contains a total of 30 path parameters. To test the robustness of our method against parameter structure, we consider two scenarios. First, we set the structure of path parameters to the same as in the reference model. This is mostly the case of subject-specific modeling, where the parameter structure is known in the generic model and calibration is performed to adjust parameter values. In the second scenario, all muscle paths were configured with either one or two cylinders and without any via points, resulting in a different parameter structure from their reference counterpart (***Appendix 2–table 2***) and approximately 10% fewer parameters in total ($\boldsymbol{p}$ and

$\boldsymbol{p}_{\text{ref}}$ in ***Equation 27*** may differ in dimension). In other words, the path is calibrated to match its moment arms with those generated by a likely more complicated muscle geometry, representing the case of generic modeling.

Apart from predetermination of parameter structure, we bounded coordinates of the origin and insertion points (${}^{\text{j}}\boldsymbol{u}_{\text{P}}$ and ${}^{\text{j}}\boldsymbol{u}_{\text{S}}$) within a uniformly distributed range of ±30 mm from the respective original values in the reference model. This is not a prerequisite for the optimization, but since the origin and insertion points attach to anatomical landmarks with accurate measurements, we included this constraint to speed up the optimization. Also, when generating the initial points, the coordinates of cylinder center (${}^{\text{j}}\boldsymbol{u}_{\text{C}}$) were randomly initialized around the sections of the line between the initial origin and insertion points. This also accelerates the process, since if the cylinder is not wrapped at the beginning of the optimization, $\partial\boldsymbol{r}/\partial{}^{\text{j}}\boldsymbol{u}_{\text{C}}$ remains a null matrix, and ${}^{\text{j}}\boldsymbol{u}_{\text{C}}$ might be left untuned till the end. Hence, we ensured that the cylinders are not too departed from the initial muscle path.

To reduce the risk of local minimum, the optimization was globally iterated with multiple initial points: For each parameter that a muscle path has (***Appendix 2–table 2***), 14 random initial points were generated; e.g., a 12-parameter path will be calibrated with at most 168 sets of different initial values. Additionally, to prevent excess computation, we programmed the optimization so that it will first iterate over 42 initial points, and if the cost (***Equation 27***) per moment arm per joint configuration does not reach a sufficiently small level k (e.g., 0.01), another 42 will be iterated. The rest of initial points will only be iterated when the normalized cost fails to reach $100k$ after the first 84 global iterations, and the calibration stops whenever $100k$ is reached or when all initial points run out. If none of the global iterations reaches the desired threshold, 14 sets of initial points that led to the lowest costs but had the solver exit due to the iteration limit are further optimized with `MaxIterations` as 1e3.

Optimization performance is also influenced by the calibration data, thus to examine the robustness of our method against error, the calibration was also performed with noisy data. For this, the aforementioned artificial moment arm data were added to with both relative error (uniformly distributed within ±20% of the reference value) and absolute error (uniformly distributed within ±2 mm).

To sum up, our method was first tested using two `lsqnonlin` algorithms with both analytical and numerical gradients, and then it was implemented for moment arm calibration in two scenarios (original and modified parameter structures) with two sets of calibration data (noise-free and noisy). This leads to a total of four conditions in the implementation, and each was tested five times with different sets of initial points to evaluate the performance in terms of calibration speed and validation accuracy. Note that the five sets of initial points were kept the same for both conditions within each scenario, and stopping criteria remained unchanged for all conditions; otherwise, the difference in results might also be attributed to initialization and configuration.

Calibration was performed on a 2.9-GHz Intel Core i9 with 64 GB RAM and 20 CPUs using parallel computing (`parfor`): the processor was not overclocked, and the number of parallel workers was set as 14. We structured parallel computing to optimize 14 initial points simultaneously for a single muscle path, which also explains why the numbers of initial points mentioned above are all common multiples of 14. Validation is assessed by mean absolute error

$$\epsilon_{\text{val}} = \frac{\sum_{n=1}^{25} |\boldsymbol{r}_{\text{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) - \boldsymbol{r}_{\text{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{opt}})|}{25}, \tag{28}$$

where $\boldsymbol{p}_{\text{opt}}$ is the solution of ***Equation 27***. Here, notice the optimized parameters are input into the non-smooth model $\boldsymbol{r}_{\text{modelGP}}$ despite obtainment from the smooth form, making the evaluation standard more strict. Furthermore, since our calibration data were generated from a reference model, we may also assess the recovery of $\boldsymbol{p}_{\text{opt}}$ against the reference values using the infinity norm (maximum absolute row sum) of parameter deviation

$$\epsilon_{\text{rec}} = \|\boldsymbol{T}_{\text{pose}}(\boldsymbol{p}_{\text{ref}})^{-1}\boldsymbol{T}_{\text{pose}}(\boldsymbol{p}_{\text{opt}})\|_{\infty}, \tag{29}$$

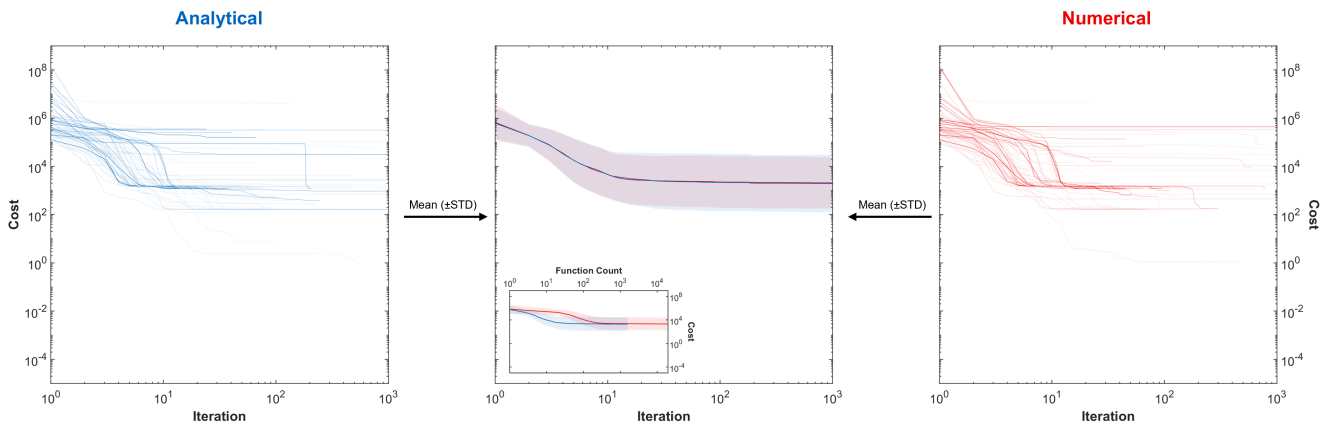


Figure 2. Cost–iteration curves for the optimization using the `levenberg-marquardt` algorithm (latissimus dorsi, iliac part). The results of each initial point from the analytical (blue) and numerical (red) gradients are plotted respectively on the left and right. The mean and STD of the costs are plotted in the middle, with function count–dependent curves in the corner.

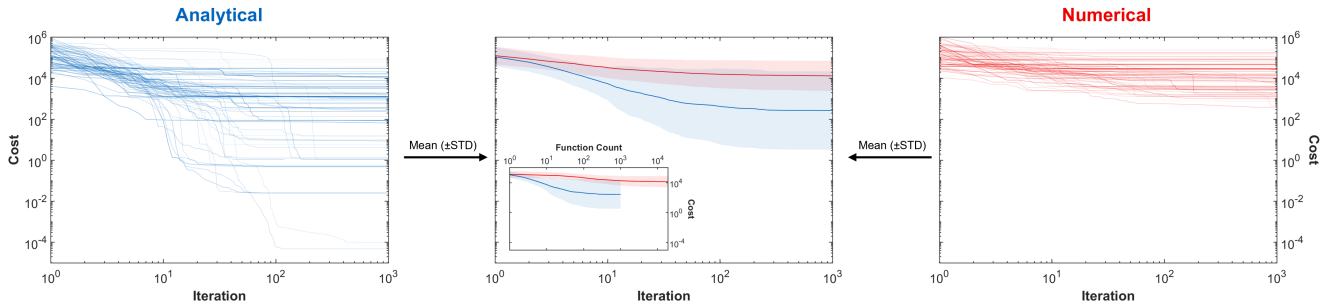


Figure 3. Cost–iteration curves for the optimization using the `trust-region-reflective` algorithm (latissimus dorsi, iliac part). The results of each initial point from the analytical (blue) and numerical (red) gradients are plotted respectively on the left and right. The mean and STD of the costs are plotted in the middle, with function count–dependent curves in the corner.

where $\boldsymbol{T}_{\text{pose}}$ is essentially a concatenated matrix of ${}^{c}\boldsymbol{T}_{j}$ from ***Equation 17***. This measure, used in relative pose estimation (***Knuth and Barooah, 2013***; ***Wang et al., 2019***; ***Guan et al., 2022***), reflects the similarity between the placements of points and cylinders configured by the reference and optimized parameters. The lower it is, the more similar the placements are, and it tends to 1 when the two sets of parameters become almost identical.

Results

We selected a few representative muscle paths to test the optimization with two classic `lsqnonlin` algorithms as well as to compare the performance with and without gradient specification. Using the `levenberg-marquardt` algorithm, the optimization process based on the analytical and numerical gradients is generally the same, with costs descending in identical fashions for most of the initial points (***Figure 2***, ***Appendix 3–figures 1–3***). Whereas with `trust-region-reflective`, the cost descents much faster with with gradient-specified optimization (***Figure 3***, ***Appendix 3–figures 4–8***): on average, by the 100th or even the 10th iteration, the cost minimized based on the analytical gradient is already lower than the final cost achieved by the numerical gradient.

We also recorded the computation time and the number of function counts (how many times the cost function is evaluated) for each iteration. ***Figure 4*** shows the ratio of computation time per iteration between the gradient-specified and unspecified optimizations, and as can be expected based on the computational principle of the numerical gradient, this ratio is almost linear to the number of parameters in the cost function. With this ratio, we scale the cost descent to the number

of function counts in ***Figures 2*** and ***3***, ***Appendix 3–figures 1–8*** to demonstrate optimization performance with respect to computation load. From this perspective, with `levenberg-marquardt`, even if the cost reduction per iteration is almost the same between the gradient-specified and unspecified optimizations, the former requires fewer function evaluations and is hence more efficient. As for the `trust-region-reflective` algorithm, the advantage achieved by the analytical gradient is further magnified by this computational efficiency.

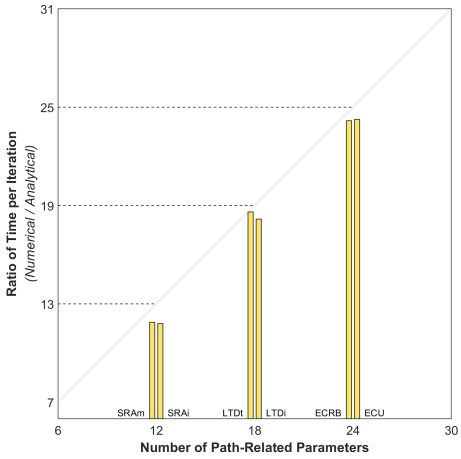


Figure 4. Ratio of computation time per iteration (numerical vs. analytical) of six representative muscle paths. See ***Appendix 2*** for abbreviations.

To demonstrate the performance of our method in implementation, we devised four conditions in which 42 muscle paths are calibrated of their multi-dimensional moment arms, and the results are summarized in ***Table 1*** and visualized in ***Figure 5***, ***Appendix 4–figures 1–3***. In an ideal condition where the parameters-to-optimize share the same structure with their reference and the calibration data contain no noise, our method completed the calibration task in 7.7 min with parallel computing. Across all 182 effective moment arms, the median validation error is 5e-5 mm and the max is 3.37 mm for the palmarflexion/dorsiflexion moment arm of the flexor carpi radialis. It is worth noticing that, as indicated by the parameter recovery assessment (***Figure 5***, bottom), about half of the optimized parameters are either identical to the reference values or proximate to them. This means that the anchor points and wrapping objects of these muscle paths are placed almost the same as in the reference musculoskeletal model, and the moment arm should be the same across all joint configurations; even if the utilized calibration dataset is not particularly large.

When the input parameters are modified of their structures (e.g., a path defined by two cylinders is now calibrated with only one cylinder, or a path with multiple via points is now calibrated without them), the calibration time is 10.9 min, and only one moment arm contains error more than 5 mm in validation. When the calibration data is noisy, the performance of our method is reduced, which is not surprising since the introduced noise is quite large (±20% relative error and ±2 mm absolute error). Nevertheless, its speed and accuracy are still satisfactory. For example, in the case of the original parameter structure, the calibration task was completed in 56.9 min with only one moment arm containing validation error over 10 mm.

For a more intuitive understanding of our method's working process, we demonstrate in ***Figures 6*** and ***7*** two cases of muscle path calibration, each with a featured complication. In ***Figure 6***,

Table 1. Method performance.

Condition		**Calibration Speed**		**Validation Accuracy**				
Parameter	Calibration	Computing Time (min)		Mean Absolute Error (mm)			Moment Arm Count*	
Structure	Data	*Serial*	*Parallel*	*Median**	*95-pct**	*Max**	*MAE >5 mm*	*MAE >10 mm*
Original	Noise-free	68.9±3.7	7.7±0.3	5e-5	1.20	3.37	0	0
Modified	Noise-free	92.4±6.3	10.9±0.9	0.09	2.62	5.58	1	0
Original	Noisy	468.6±8.1	56.9±0.4	1.87	6.74	10.17	19	1
Modified	Noisy	340.0±6.9	45.3±0.5	1.79	6.09	10.50	17	2

*Calculated or counted from the 182 effective moment arms of all 42 muscle paths.

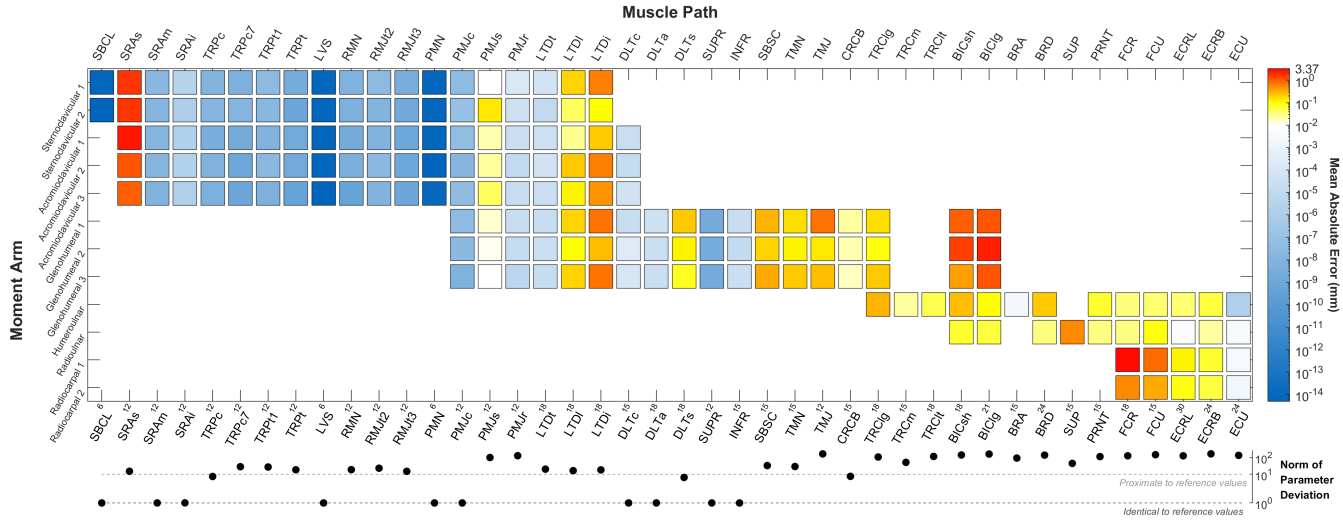


Figure 5. Absolute errors in 182 effective moment arms of 42 muscle paths in validation (original parameter structure and noise-free calibration data). The magnitude of the error is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle path. See ***Appendix 2*** for nomenclatures in the axes.

both the initial origin and insertion points are within the cylinder. This is a situation that the original obstacle-set method cannot handle, and we patched the issue by shifting such point onto to the cylinder surface to prevent a breakdown in wrapping point computation. As soon as the optimization starts, they are gradually expelled out of the cylinder to reduce the cost, achieving a smooth transition from one branch to another within a conditional statement. ***Figure 7*** shows a typical path configured by two cylinders, and the misplacement of either cylinder will induce error in moment arm computation. Driven by the gradient, first the orientations of both cylinders are rotated to the correct directions, and then their radii are adjusted to the right sizes; all in tens of iterations.

Discussion

A key to optimization speed and accuracy is the gradient, which in our case is the gradient of the moment arm function with path-related parameters as variables. To derive it analytically, the cost function must be differentiable in the first place. For this, we revised the obstacle-set method for muscle path wrapping by substituting its conditional statements and nondifferentiable components with soft functions. Next, we disassembled the revised function into smaller modules that are easier to derive separately, and then assembled the gradients back into the desired gradient composite. The result is simplistic in form and thus easy to compute, which further increases optimization speed. Importantly, the concept of the chain rule is universal and may be useful for deriving the gradient of other complex functions as well.

Specifying the gradient in its analytical form is more than a formal alternative to gradient approximation using the typical finite difference method. As we demonstrate in our preliminary tests, the performance of the gradient-specified optimization is distinct from that of the gradient-unspecified. This arises from how the analytical and numerical gradients are computed as well as how each optimization algorithm proceeds with different gradients. For example, to estimate the gradient numerically with the forward-difference method, consider the equation

$$\frac{\partial J(\boldsymbol{p})}{\partial \boldsymbol{p}} \approx \left[\frac{J(\boldsymbol{p}+\delta \mathbf{e}_1)-J(\boldsymbol{p})}{\delta} \quad \frac{J(\boldsymbol{p}+\delta \mathbf{e}_2)-J(\boldsymbol{p})}{\delta} \quad \dots \quad \frac{J(\boldsymbol{p}+\delta \mathbf{e}_n)-J(\boldsymbol{p})}{\delta} \right], \tag{30}$$

where $\mathbf{e}_i$ is the standard basis vector with the i-th element as 1 and 0s elsewhere, and δ is a sufficiently small positive value. To approximate the gradient, the additional number of function evaluations equals to the dimension of input parameters; this number will double if a central-difference

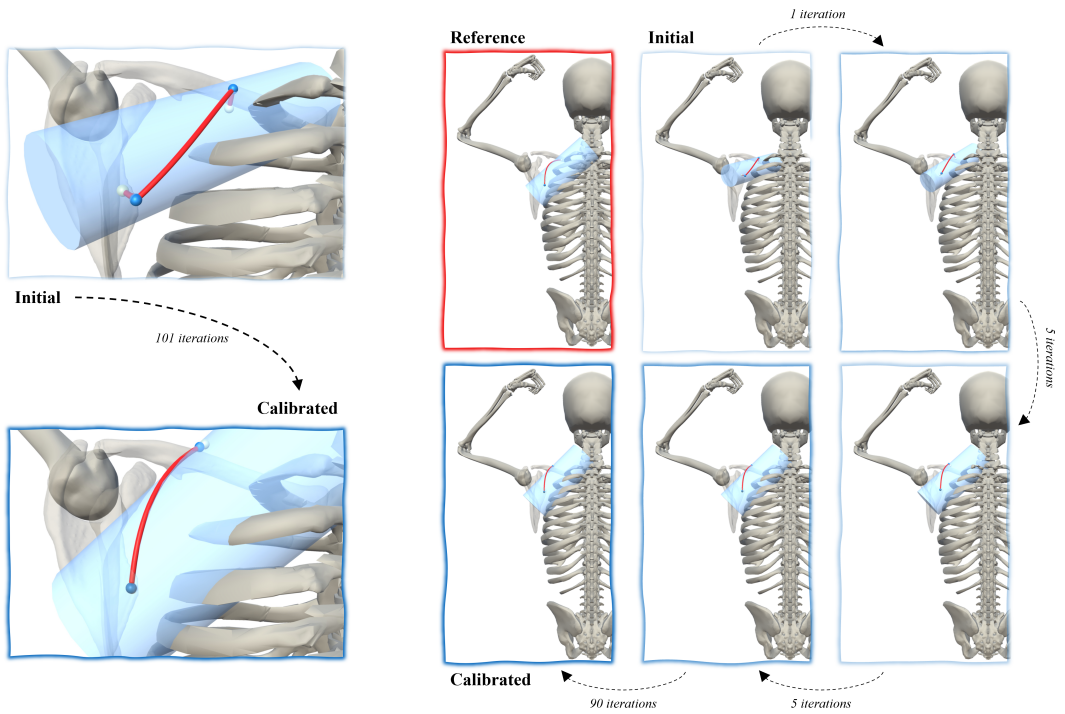


Figure 6. A featured case of path calibration for the serratus anterior (superior part). Left: anchor points and wrapping obstacles defined by pre- and post-optimized parameters. Right: the reference path configuration and the optimization process from initial point to solution. The yellow points on the red muscle path are the origin and insertion, while the wrapping points are in blue. At the initial point, the origin and insertion are inside the wrapping obstacle, which would induce a computation breakdown in the original obstacle-set algorithm but is patched by our revision. As the optimization progresses, the origin and insertion are gradually expelled from the obstacle, leading to a path configuration valid for the original algorithm.

method is used. In other words, the time complexity of ***Equation 30*** is $O(\dim(\boldsymbol{p}))$ (***Hopcroft and Rus, 2003***; ***Papadimitriou, 2003***; ***Oliveto et al., 2007***). However, if the analytical form is explicit, the cost function only needs to be evaluated once, since most terms for computing the gradient analytically are already accessible as part of the cost function evaluation; time complexity being $O(1)$. As shown in ***Figure 4***, the average computation time per iteration based on the numerical gradient is approximately ($\dim(\boldsymbol{p})$+1) times of that based on the analytical, so the gradient-specified optimization is always much faster than the unspecified.

As for how the optimization progresses each iteration, the specifics depend on the algorithm. For instance, the `levenberg-marquardt` algorithm has only one conditional statement, in which a search direction–related factor is either magnified or reduced based on if a step (i.e., an iteration) succeeds in reducing the cost (***The MathWorks Inc., 2024***). Consequently, the residual between the analytical and numerical gradients will not induce any difference in the optimization, unless the accumulated numerical error happens to diverge a step from success to failure. For most of the initial points we tested, the optimization process is identical between the analytical and numerical gradients (***Figure 2***, ***Appendix 3–figures 1–3***). This also verifies our derivation of the analytical gradient.

With the `trust-region-reflective` algorithm, there is a much larger difference in the results between the gradient-specified and unspecified optimizations (***Figure 3***, ***Appendix 3–figures 4–8***). The reason is that this algorithm not only contains a conditional statement with three branches that could separate the optimization process, but these branches are unique in terms of determining the direction of a search step: When the Hessian of the cost function is positive definite, the Newton's method is applicable for rapid local convergence (namely a step in the Newton direc-

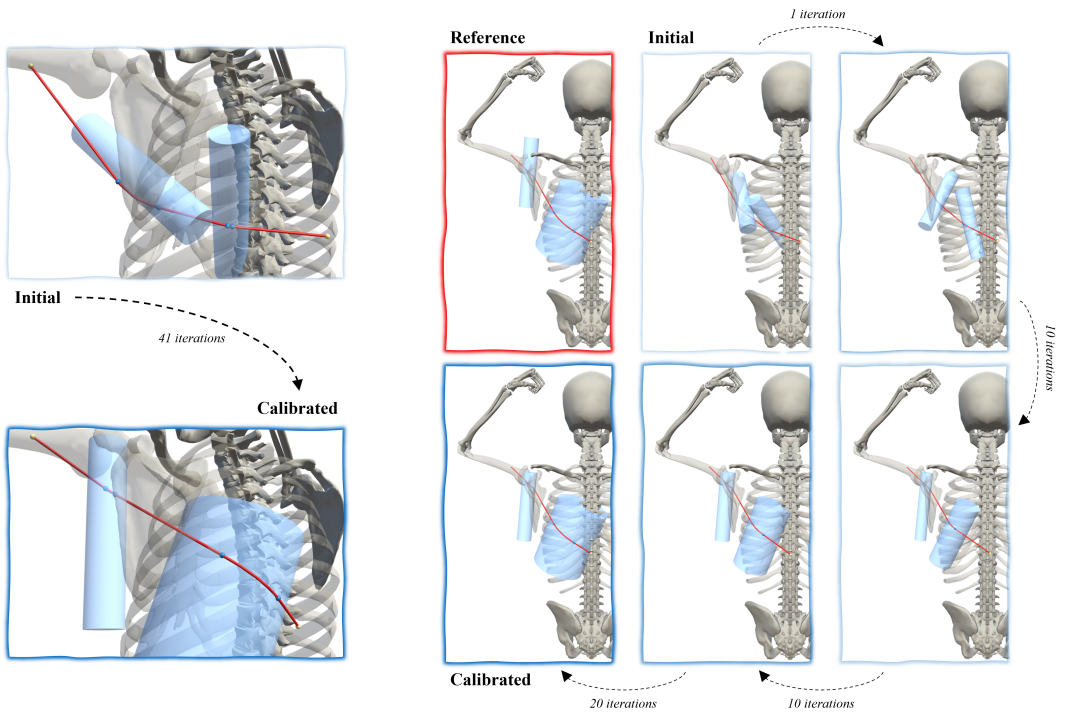


Figure 7. A featured case of path calibration for the latissimus dorsi (thoracic part). Left: anchor points and wrapping obstacles defined by pre- and post-optimized parameters. Right: the reference path configuration and the optimization process from initial point to solution. The yellow points on the red muscle path are the origin and insertion, while the wrapping points are in blue. The path is complicated in that it wraps around two obstacles, both of which need to be correctly placed for moment arm calculation to be accurate. By the 11th iteration, the cylinders are oriented to the correct directions, and their radii are adjusted correctly in another 30 iterations. The parameters related to cylinder placement are optimized without constraints.

tion), otherwise the solution must be approached either in the direction of the steepest descent or negative curvature (***Nocedal and Wright, 2006***; ***The MathWorks Inc., 2024***). Since the Hessian is approximated using quasi-Newton methods (***Nocedal and Wright, 2006***), if the gradient itself is already a numerical estimation, the Hessian approximation might not be positive definitive even if its exact value is. In this case, a Newton step is no longer possible, and neither of the alternatives converges to the solution as quickly. With the analytical gradient, however, the approximation is closer to the exact Hessian. Based on our observation so far, for every iteration, the Hessian approximated based on the analytical gradient maintains positive definite, indicating that Newton steps are likely taken throughout the optimization even with alternatives in `trust-region-reflective`, hence rapid convergence is ensured. Notably, the positive definiteness of the Hessian also allows the application of conjugate gradient methods to obtain each Newton step (***Nocedal and Wright, 2006***), avoiding the need for inverting the Hessian using Gaussian elimination, which is computationally less efficient.

To encapsulate, specifying the gradient in its analytical form benefits the optimization in two ways:

1. efficient computation of every iteration, and
2. rapid convergence (i.e., fewer iterations) in trust-region methods.

These two advantages grant the gradient-specified optimization much more cost descent per function count compared with the unspecified, allowing the search from many more initial points for the global solution. For instance, this rapidly convergent iteration process can be utilized for selection and mutation in genetic algorithm (***Forrest, 1993***; ***Oliveto et al., 2007***; ***Mirjalili, 2019***) and guarantees producing of the locally optimal child without generating a large population.

With these advantages, the specification of gradient resolves most difficulties for the optimization in practice, but not all of them. Ideally, had the solver run for enough time with a sufficiently large number of iterations, there is not much need for any further conditions or constraints. However, expenditures in computation and time are always a concern in reality, and we may as well include some conditions as long as the required information is not more than we already have.

To begin with, the type of obstacle as well as the number are predetermined in our method to define the vector structure of path parameters. It is of course possible to introduce a soft function that merges moment arm computation with different amount of various obstacles, so that the obstacle type and number can also be optimized. Yet we find it unnecessary at the current stage, since the cylinder is a more popular obstacle for muscle path modeling, and the obstacle number is highly related to the number of joints a muscle crosses so it can be easily predetermined.

We also constrained the 3-D coordinates of the origin and insertion points to a range (±30 mm) around the reference values; imagine the volume of a standard Rubik's Cube. This is not essential for our method, but it is also not necessary to spend time optimizing what is already quite clear. The origin and insertion are attached to anatomical landmarks, whose relative locations on the bones are well-studied. Especially if the goal is to calibrate upper-limb muscle path, even with individual variability taken into account, the origin and insertion points should not shift too much from our selected bounds. Other parameters, particularly those of the cylinders, were tuned without any constraint, so the major complication of obstacle placement is still being tackled without the knowledge of musculoskeletal geometry (***Figure 7***). In fact, our modification of the obstacle-set method has already removed complex geometric constraints between the cylinder and the points, and the optimization may proceed from any initial point (***Figure 6***).

Last, there is the typical problem of local minimum, which cannot be avoided even with the optimization gradient and constraints. For this, we adopted the common strategy of multiple initial points, and for each muscle, 84–420 different initial points could be iterated in optimization. This does not eliminate the risk of local minimum, but should avoid it as much as possible. Again, it would be neat and tempting to hit the global minimum with only one shot, but our approach is more practical with the computational cost taken into account.

The performance of our method is demonstrated by replicating the muscle paths of a reference model in four conditions (***Table 1***). Here, note that we validated the optimized parameters using non-smooth wrapping and moment arm computation, meaning that these parameters are (sub-)optimal for the original model as well. It is hence practical to implement our method for other models: For example, with some slight modifications to accommodate other wrapping strategies (not necessary if the outcome is similar), it is possible to build upon our method a user-friendly program, which takes in any desired skeleton model and moment arm–joint angle relations for muscle path calibration reproducible in OpenSim and other software.

Moreover, we need to reiterate that it is mainly the high dimensional relation between moment arm and joint configuration that makes muscle path calibration challenging, which is often neglected. The moment arm in our reference model is not a one-dimensional variable but a vector of 12 elements. For instance, the pectoralis major crosses the shoulder complex and actuates up to eight DoFs, and each of the accordant moment arms varies with changes in any one of the eight DoFs, meaning that their moment arm–joint angle relation is depicted in a 8-D space. Calibrating the moment arms of the pectoralis major is similar to matching eight pairs of 8-D hypersurfaces, and it is almost impossible to accomplish manually unless four or five moment arms are neglected. In spite of that, our method succeeds in calibrating most muscles with satisfactory speed and accuracy: e.g., the three parts of the pectoralis major were each calibrated in a few seconds, and the validation errors in their 24 moment arms are all negligible.

One limitation of this study is that the performance of our method is demonstrated only with artificial data generated from a model. To eliminate the concern that our optimization performance mainly benefits from the same structure shared by the parameters in the reference model and those to be calibrated, we also tested the scenario where a simpler parameter structure is config-

ured. As shown in ***Table 1*** and ***Appendix 4–figure 1***, though both calibration speed and validation accuracy decrease compared with when the original parameter structure is configured, the calibration time is only 10.9 min and the median validation error is 0.09 mm. In reality, the path of a muscle is geometrically far more complicated than a few points and wrapping obstacles, but our results show that the optimization could still succeed with a relatively simple parameter structure. We also tested conditions with noisy data, and our method is quite robust (***Table 1***, ***Appendix 4–figures 2*** and ***3***) against relative and absolute errors likely larger than expected in experimental measurements. Importantly, our method is conveniently compatible with experimental data, since the only mandatory input is the relationship between moment arms and joint angles, regardless of how the data are obtained or whether the subject is human.

Another expected concern could be that, due to current technical limits, we usually do not have access to the measurements for all moment arms of a muscle, let alone their relations with all actuating DoFs. Yet this is not a problem related to our method but rather faced by all biomechanists. When lacking measurements, the moment arms will simply be assumed of certain values or relations, which is essentially what has been done in musculoskeletal modeling. Nevertheless, had such advanced measurements been available, our method is in line to provide equally advanced calibration. In fact, even if the high dimensional moment arm–joint angle relation is not completely established, our method is already usefully applicable. For example, to calibrate the Achilles tendon plantarflexion moment arm for a subject-specific model, one may use ultrasound or MRI to perform the measurement at the neutral ankle position, and then linearly scale the well-investigated generic moment arm curves (***Maganaris et al., 2000***; ***Ward et al., 2009***; ***Fath et al., 2010***; ***Holzer et al., 2020***) to obtain $\boldsymbol{r}_{\text{target}}(\boldsymbol{q})$ for our method. Or if one assumes some relation between moment arm and subject anthropometry, then they can also take moment arm values in the generic model as reference, scale them based on the assumed relation, and input in our method to generate a subject-specific model. In either case, the resultant muscle path will be much more accurate than when path parameters are directly scaled based on skeletal geometry.

Given these limitations, we expect our method to be examined with more models and datasets. As aforementioned, a practical application of this method would be in subject-specific modeling to recalibrate muscle path after scaling, which will be demonstrated in our future work. We are currently utilizing this method to develop a lower-limb model based on experimental data, which should offer further validation aligned with the theoretics in the current study.

It is also worth noticing that, apart from the key role in muscle path calibration, our analytically derived gradient reveals conclusively the sensitivity of moment arm to path-related parameters. The value of each element in the gradient denotes how much influence each parameter has on moment arm in a certain joint configuration, which could be of important reference in medicine. For example, in the surgery repairing rotator cuff tears, the positional error of tendon reattachment or graft insertion would induce unwanted changes in moment arms, whose magnitudes may be different depending on the direction of positional shift. The information of sensitivity could help the surgeons or manufacturers of medical devices to focus on limiting the error in the direction with potentially larger moment arm variance, so as not to compromise the biomechanical function of the operated shoulder.

Finally, we recapitulate the universal practicability of the concept we employed in the derivation of optimization gradient. A complex function may be disassembled into a product of simple modules for separate derivation, and the gradients of each module not only can be reassembled into the desired gradient composite, but may also be used as part of the gradient in other cost functions. A good example would be $\partial\boldsymbol{u}/\partial\boldsymbol{p}$ in ***Equation 15***, from which we conveniently derived the gradient for a path coordinate–based cost function. This enables calibration with medical imaging data (e.g., MRI) to expand the applicability. Furthermore, it would be interesting if kinetic measurements such as joint moment can be utilized in calibration, which requires the optimization of musculotendon parameters. In a previous study, we have analytically derived the requisite gradient (***Chen and Franklin, 2023***), and it can be integrated with this work, making possible joint

moment–based model calibration, which is key to automated musculoskeletal modeling.

Acknowledgments

This work was supported by the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant no. 5140951) and the German Federal Ministry of Education and Research funding (Project AI.D, grant no. 16ME0539K).

References

Ao D, Li G, Shourijeh MS, Patten C, Fregly BJ. EMG-driven musculoskeletal model calibration with wrapping surface personalization. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2023; 31:4235–4244. doi: 10.1109/TNSRE.2023.3323516.

Aper RL, Saltzman CL, Brown TD. The effect of hallux sesamoid excision on the flexor hallucis longus moment arm. Clinical Orthopaedics and Related Research. 1996; 325:209–217. doi: 10.1097/00003086-199604000-00025.

Arnold AS, Salinas S, Hakawa DJ, Delp SL. Accuracy of muscle moment arms estimated from MRI-based musculoskeletal models of the lower extremity. Computer Aided Surgery. 2000; 5(2):108–119. doi: 10.3109/10929080009148877.

Arnold EM, Ward SR, Lieber RL, Delp SL. A model of the lower limb for analysis of human movement. Annals of Biomedical Engineering. 2010; 38(2):269–279. doi: 10.1007/s10439-009-9852-5.

Chen Z, Franklin DW. Musculotendon parameters in lower limb models: Simplifications, uncertainties, and muscle force estimation sensitivity. Annals of Biomedical Engineering. 2023; 51(6):1147–1164. doi: 10.1007/s10439-023-03166-5.

Delp SL, Anderson FC, Arnold AS, Loan P, Habib A, John CT, Guendelman E, Thelen DG. OpenSim: Open-source software to create and analyze dynamic simulations of movement. IEEE Transactions on Biomedical Engineering. 2007; 54(11):1940–1950. doi: 10.1109/TBME.2007.901024.

Delp SL, Hess WE, Hungerford DS, Jones LC. Variation of rotation moment arms with hip flexion. Journal of Biomechanics. 1999; 32(5):493–501. doi: 10.1016/S0021-9290(99)00032-9.

Delp SL. Surgery simulation: A computer graphics system to analyze and design musculoskeletal reconstructions of the lower limb. Doctoral dissertation, Stanford University; 1990.

Fath F, Blazevich AJ, Waugh CM, Miller SC, Korff T. Direct comparison of in vivo Achilles tendon moment arms obtained from ultrasound and MR scans. Journal of Applied Physiology. 2010; 109(6):1644–1652. doi: 10.1152/japplphysiol.00656.2010.

Forrest S. Genetic algorithms: Principles of natural selection applied to computation. Science. 1993; 261(5123):872–878. doi: 10.1126/science.8346439.

Garner BA, Pandy MG. The obstacle-set method for representing muscle paths in musculoskeletal models. Computer Methods in Biomechanics and Biomedical Engineering. 2000; 3(1):1–30. doi: 10.1080/10255840008915251.

Garner BA, Pandy MG. Musculoskeletal model of the upper limb based on the Visible Human Male dataset. Computer Methods in Biomechanics and Biomedical Engineering. 2001; 4(2):93–126. doi: 10.1080/10255840008908000.

Gollapudi SK, Lin DC. Experimental determination of sarcomere force–length relationship in type-I human skeletal muscle fibers. Journal of Biomechanics. 2009; 42(13):2011–2016. doi: 10.1016/j.jbiomech.2009.06.013.

Gordon AM, Huxley AF, Julian FJ. The variation in isometric tension with sarcomere length in vertebrate muscle fibres. The Journal of Physiology. 1966; 184(1):170–192. doi: 10.1113/jphysiol.1966.sp007909.

Guan B, Zhao J, Li Z, Sun F, Fraundorfer F. Relative pose estimation with a single affine correspondence. IEEE Transactions on Cybernetics. 2022; 52(10):10111–10122. doi: 10.1109/TCYB.2021.3069806.

Guo J, Huang H, Yu Y, Liang Z, Ambrósio J, Zhao Z, Ren G, Ao Y. Modeling muscle wrapping and mass flow using a mass-variable multibody formulation. Multibody System Dynamics. 2020; 49(3):315–336. doi: 10.1007/s11044-020-09733-1.

Herzog W, ter Keurs HEDJ. A method for the determination of the force–length relation of selected in-vivo human skeletal muscles. Pflügers Archiv. 1988; 411(6):637–641. doi: 10.1007/BF00580859.

Hicks JL, Uchida TK, Seth A, Rajagopal A, Delp SL. Is my model good enough? Best practices for verification and validation of musculoskeletal models and simulations of movement. Journal of Biomechanical Engineering. 2015; 137(2):020905. doi: 10.1115/1.4029304.

Hintermann B, Nigg BM, Sommer C. Foot movement and tendon excursion: An in vitro study. Foot & Ankle International. 1994; 15(7):386–395. doi: 10.1177/107110079401500708.

Holzbaur KRS, Murray WM, Delp SL. A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control. Annals of Biomedical Engineering. 2005; 33(6):829–840. doi: 10.1007/s10439-005-3320-7.

Holzer D, Paternoster FK, Hahn D, Siebert T, Seiberl W. Considerations on the human Achilles tendon moment arm for in vivo triceps surae muscle–tendon unit force estimates. Scientific Reports. 2020; 10(1):19559. doi: 10.1038/s41598-020-76625-x.

Hopcroft JE, Rus D. Algorithms, analysis of. In: *Encyclopedia of Computer Science*; 2003.p. 42–45.

Hu T, Kuehn J, Haddadin S. Identification of human shoulder-arm kinematic and muscular synergies during daily-life manipulation tasks. In: *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics*; 2018. p. 1011–1018. doi: 10.1109/BIOROB.2018.8487190.

Hu T, Kühn J, Haddadin S. Forward and inverse dynamics modeling of human shoulder–arm musculoskeletal system with scapulothoracic constraint. Computer Methods in Biomechanics and Biomedical Engineering. 2020; 23(11):785–803. doi: 10.1080/10255842.2020.1764945.

Kane TR, Levinson DA. Dynamics: Theory and applications. McGraw-Hill; 1985.

Killen BA, Brito da Luz S, Lloyd DG, Carleton AD, Zhang J, Besier TF, Saxby DJ. Automated creation and tuning of personalised muscle paths for OpenSim musculoskeletal models of the knee joint. Biomechanics and Modeling in Mechanobiology. 2021; 20(2):521–533. doi: 10.1007/s10237-020-01398-1.

Knuth J, Barooah P. Error growth in position estimation from noisy relative pose measurements. Robotics and Autonomous Systems. 2013; 61(3):229–244. doi: 10.1016/j.robot.2012.11.001.

Lee SSM, Piazza SJ. Inversion–eversion moment arms of gastrocnemius and tibialis anterior measured in vivo. Journal of Biomechanics. 2008; 41(16):3366–3370. doi: 10.1016/j.jbiomech.2008.09.029.

Lloyd JE, Roewer-Després F, Stavness I. Muscle path wrapping on arbitrary surfaces. IEEE Transactions on Biomedical Engineering. 2021; 68(2):628–638. doi: 10.1109/TBME.2020.3009922.

Maganaris CN, Baltzopoulos V, Sargeant AJ. In vivo measurement-based estimations of the human Achilles tendon moment arm. European Journal of Applied Physiology. 2000; 83(4-5):363–369. doi: 10.1007/s004210000247.

Ma'touq J, Hu T, Haddadin S. A validated combined musculotendon path and muscle-joint kinematics model for the human hand. Computer Methods in Biomechanics and Biomedical Engineering. 2019; 22(7):727–739. doi: 10.1080/10255842.2019.1588256.

McCullough MBA, Ringleb SI, Arai K, Kitaoka HB, Kaufman KR. Moment arms of the ankle throughout the range of motion in three planes. Foot & Ankle International. 2011; 32(3):300–306. doi: 10.3113/FAI.2011.0300.

McFarland DC, Binder-Markey BI, Nichols JA, Wohlman SJ, de Bruin M, Murray WM. A musculoskeletal model of the hand and wrist capable of simulating functional tasks. IEEE Transactions on Biomedical Engineering. 2023; 70(5):1424–1435. doi: 10.1109/TBME.2022.3217722.

Millard M, Franklin DW, Herzog W. A three filament mechanistic model of musculotendon force and impedance. eLife. 2024; 12:RP88344. doi: 10.7554/eLife.88344.4.

Millard M, Uchida T, Seth A, Delp SL. Flexing computational muscle: Modeling and simulation of musculotendon dynamics. Journal of Biomechanical Engineering. 2013; 135(2):021005. doi: 10.1115/1.4023390.

Mirjalili S. Genetic Algorithm. In: *Evolutionary Algorithms and Neural Networks: Theory and Applications*; 2019.p. 43–55. doi: 10.1007/978-3-319-93025-1_4.

Modenese L, Kohout J. Automated generation of three-dimensional complex muscle geometries for use in personalised musculoskeletal models. Annals of Biomedical Engineering. 2020; 48(6):1793–1804. doi: 10.1007/s10439-020-02490-4.

Nocedal J, Wright SJ. Numerical optimization. Springer New York; 2006. doi: 10.1007/978-0-387-40065-5.

Oliveto PS, He J, Yao X. Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results. International Journal of Automation and Computing. 2007; 4(3):281–293. doi: 10.1007/s11633-007-0281-3.

Papadimitriou CH. Computational complexity. In: *Encyclopedia of Computer Science*; 2003.p. 260–265.

Rajagopal A, Dembia CL, DeMers MS, Delp DD, Hicks JL, Delp SL. Full-body musculoskeletal model for muscle-driven simulation of human gait. IEEE Transactions on Biomedical Engineering. 2016; 63(10):2068–2079. doi: 10.1109/TBME.2016.2586891.

Scheys L, Spaepen A, Suetens P, Jonkers I. Calculated moment-arm and muscle-tendon lengths during gait differ substantially using MR based versus rescaled generic lower-limb musculoskeletal models. Gait & Posture. 2008; 28(4):640–648. doi: 10.1016/j.gaitpost.2008.04.010.

Sherman MA, Seth A, Delp SL. What is a moment arm? Calculating muscle effectiveness in biomechanical models using generalized coordinates. In: *9th International Conference on Multibody Systems, Nonlinear Dynamics, and Control*; 2013. p. V07BT10A052. doi: 10.1115/DETC2013-13633.

The MathWorks Inc , Least-squares (model fitting) algorithms documentation; 2024. https://www.mathworks.com/help/optim/ug/least-squares-model-fitting-algorithms.html. The MathWorks Inc.

Vaarbakken K, Steen H, Samuelsen G, Dahl HA, Leergaard TB, Nordsletten L, Stuge B. Lengths of the external hip rotators in mobilized cadavers indicate the quadriceps coxa as a primary abductor and extensor of the flexed hip. Clinical Biomechanics. 2014; 29(7):794–802. doi: 10.1016/j.clinbiomech.2014.05.011.

Wang S, Jia Q, Chen G, Liu D. Complete relative pose error model for robot calibration. Industrial Robot: The International Journal of Robotics Research and application. 2019; 46(5):622–630. doi: 10.1108/IR-11-2018-0244.

Ward SR, Eng CM, Smallwood LH, Lieber RL. Are current measurements of lower extremity muscle architecture accurate? Clinical Orthopaedics and Related Research. 2009; 467(4):1074–1082. doi: 10.1007/s11999-008-0594-8.

Weinstein JN, Andriacchi TP, Galante JO. A relationship of total knee design kinematics to patient function. The Iowa Orthopaedic Journal. 1987; 7:85–90.

Wolfram S, Morse CI, Winwood KL, Hodson-Tole E, McEwan IM. Achilles tendon moment arm in humans is not affected by inversion/eversion of the foot: A short report. Royal Society Open Science. 2018; 5(1):171358. doi: 10.1098/rsos.171358.

Wretenberg P, Németh G, Lamontagne M, Lundin B. Passive knee muscle moment arms measured in vivo with MRI. Clinical Biomechanics. 1996; 11(8):439–446. doi: 10.1016/S0268-0033(96)00030-7.

Yamaguchi GT. Dynamic modeling of musculoskeletal motion: A vectorized approach for biomechanical analysis in three dimensions. Springer US; 2001. doi: 10.1007/978-0-387-28750-8.

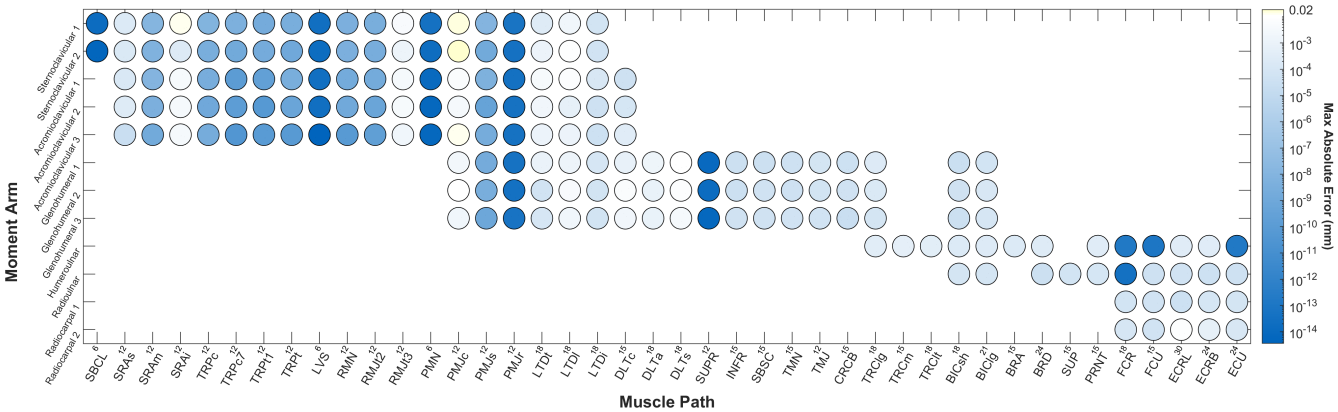
Appendix 1

The smooth modification of the obstacle-set method (***Garner and Pandy, 2000***) and moment arm computation is verified in a way similar to the validation process described in the *Calibration and Validation* subsection

$$\epsilon_{\text{ver}} = \max_{n=1}^{1000} |\boldsymbol{r}_{\text{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) - \boldsymbol{r}_{\text{modelGP+}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}})|, \tag{31}$$

where $\boldsymbol{r}_{\text{modelGP}}$ is based on the slightly revised obstacle-set method (***Figure 1***, middle) and $\boldsymbol{r}_{\text{modelGP+}}$ is the smooth form described in the *Soft Functions* subsection. The input $\boldsymbol{p}_{\text{ref}}$ is path-related parameters from ***Garner and Pandy*** (***2001***), and $\boldsymbol{q}_n$ is interpolated from the kinematic measurements of the shoulder and arm from 10 intransitive daily tasks (100 joint configurations per task) performed by a single subject in ***Hu et al.*** (***2018***), including gesturing an OK sign, pumping fists, blocking out light from the face, saluting, pointing, gesturing a thumb-down, signaling for hitchhike, greeting, gesturing to stop, and gesturing for silence.

Moment arm is computed in a 12-DoF human shoulder–arm model (***Hu et al., 2020***); see ***Appendix 2–table 1*** for DoFs in detail. As shown in the figure below, across 1000 joint configurations in the range of motion, the moment arm computed based on our smoothed method has negligible difference from the original value, with the maximum from 182 effective moment arms of 42 muscle paths being 0.02 mm; see ***Appendix 2–table 2*** for muscle paths in detail. The existing difference stems from the nature of the soft functions, rather than coding issues.



Appendix 1—figure 1. Absolute errors in 182 effective moment arms of 42 muscle paths (max value across 1000 joint configurations). The magnitude of the error is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle. See ***Appendix 2*** for nomenclatures in the axes.

720 ## Appendix 2

Appendix 2—table 1. Nomenclature for 12 DoFs in the human shoulder and arm (***Hu et al., 2020***).

Joint	DoF (Moment Arm)
Sternoclavicular 1	Protraction/retraction
Sternoclavicular 2	Elevation/depression
Acromioclavicular 1	Lateral abduction/adduction
Acromioclavicular 2	Internal/external rotation
Acromioclavicular 3	Posterior abduction/adduction
Glenohumeral 1	Anteversion/retroversion
Glenohumeral 2	Abduction/adduction
Glenohumeral 3	Interior/exterior rotation
Humeroulnar	Flexion/extension
Radioulnar	Pronation/supnation
Radiocarpal 1	Palmarflexion/dorsiflexion
Radiocarpal 2	Radial/ulnar abduction

Appendix 2—table 2. Nomenclature and parameter structure for 42 muscle paths in the human shoulder and arm.
Parameter structure is denoted by a combination of the origin (O), via (V), and insertion (I) points, as well as cylinders (C) in the specified order. The number of parameters is shown in the bracket following each structure. The original structure is defined by ***Garner and Pandy*** (***2001***).

Acronym	Muscle Path	Parameter Structure	
		Original	Modified
SBCL	subclavius	OI (6)	OCI (12)
SRAs	serratus anterior (superior)	OCI (12)	OCI (12)
SRAm	serratus anterior (middle)	OCI (12)	OCI (12)
SRAi	serratus anterior (inferior)	OCI (12)	OCI (12)
TRPc	trapezius (C1–C6)	OCI (12)	OCI (12)
TRPc7	trapezius (C7)	OCI (12)	OCI (12)
TRPt1	trapezius (T1)	OCI (12)	OCI (12)
TRPt	trapezius (T2–T7)	OCI (12)	OCI (12)
LVS	levator scapulae	OI (6)	OCI (12)
RMN	rhomboid minor	OCI (12)	OCI (12)
RMJt2	rhomboid major (T1–T2)	OCI (12)	OCI (12)
RMJt3	rhomboid major (T3–T4)	OCI (12)	OCI (12)
PMN	pectoralis minor	OI (6)	OCI (12)
PMJc	pectoralis major (clavicularis)	OCI (12)	OCI (12)
PMJs	pectoralis major (sternocostalis)	OCI (12)	OCI (12)
PMJr	pectoralis major (abdominalis)	OCI (12)	OCI (12)
LTDt	latissimus dorsi (thoracic)	OCCI (18)	OCCI (18)
LTDl	latissimus dorsi (lumbar)	OCCI (18)	OCCI (18)
LTDi	latissimus dorsi (iliac)	OCCI (18)	OCCI (18)
DLTc	deltoid (clavicular)	OCVI (15)	OCCI (18)
DLTa	deltoid (acrominal)	OVCVI (18)	OCCI (18)
DLTs	deltoid (scapular)	OVCVI (18)	OCCI (18)
SUPR	supraspinatus	OVVI (12)	OCCI (18)
INFR	infraspinatus	OVCI (15)	OCI (12)
SBSC	subscapularis	OVCI (15)	OCI (12)
TMN	teres minor	OVCI (15)	OCI (12)

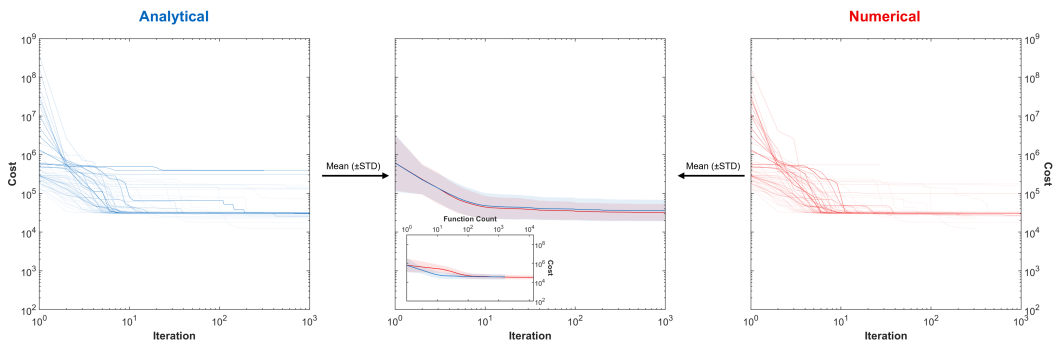
Acronym	Muscle Path	Parameter Structure	
		Original	Modified
TMJ	teres major	OCI (12)	OCI (12)
CRCB	coracobrachialis	OCVI (15)	OCI (12)
TRClg	triceps brachii (long head)	OCCI (18)	OCCI (18)
TRCm	triceps brachii (medial)	OVCI (15)	OCI (12)
TRClt	triceps brachii (lateral)	OVCVI (18)	OCI (12)
BICsh	biceps (short head)	OCCI (18)	OCCI (18)
BIClg	biceps (long head)	OCVCI (21)	OCCI (18)
BRA	brachialis	OVCI (15)	OCI (12)
BRD	brachioradialis	OVCCVI (24)	OCCI (18)
SUP	supinator	OCVI (15)	OCI (12)
PRNT	pronator teres	OVCI (15)	OCI (12)
FCR	flexor carpi radialis	OVCVI (18)	OCI (12)
FCU	flexor carpi ulnaris	OVCI (15)	OCI (12)
ECRL	extensor carpi radialis longus	OVCVVCVI (30)	OCCI (18)
ECRB	extensor carpi radialis brevis	OCVCVI (24)	OCCI (18)
ECU	extensor carpi ulnaris	OVVVCCI (24)	OCI (12)

Appendix 3

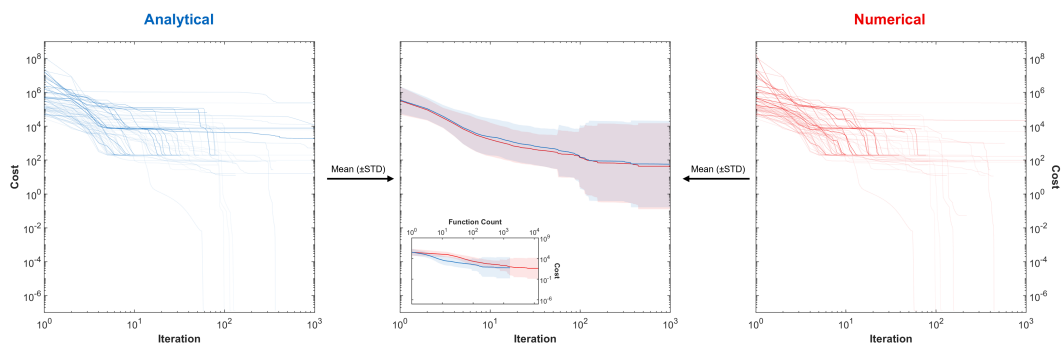
The following figures are supplements to ***Figure 2*** and ***Figure 3***, showing the cost–iteration curves for the optimization using either the algorithm of

1. `levenberg-marquardt` (70 initial points), or
2. `trust-region-reflective` (84 initial points).

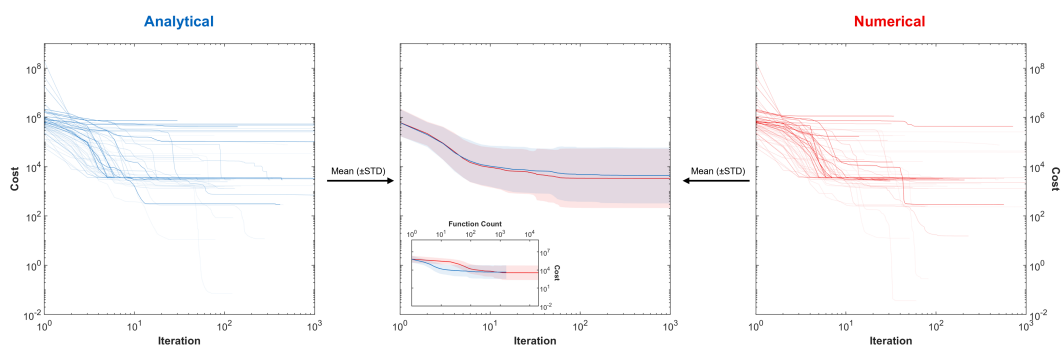
The results of each initial point from the analytical (blue) and numerical (red) gradients are plotted respectively on the left and right. The mean and STD of the costs are plotted in the middle, with function count–dependent curves in the corner.



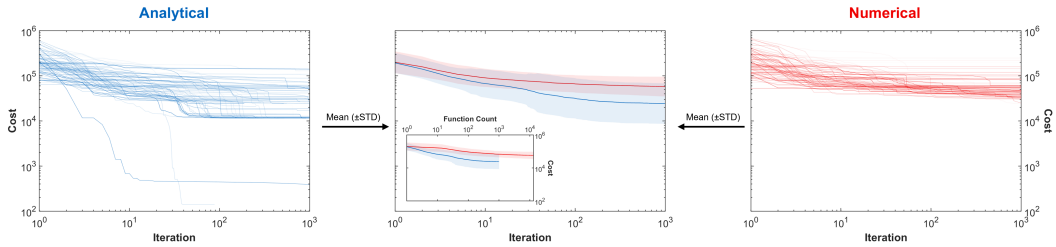
Appendix 3—figure 1. `levenberg-marquardt`, serratus anterior (superior part).



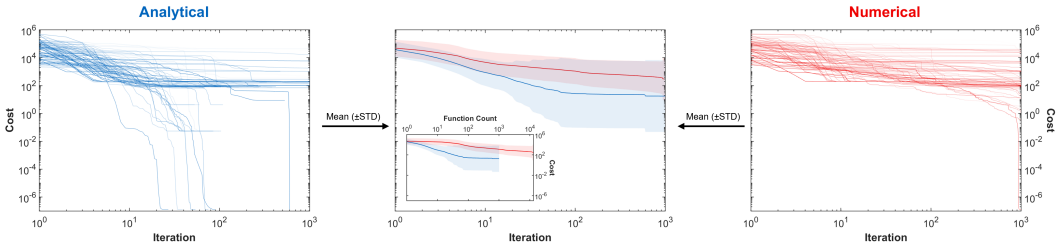
Appendix 3—figure 2. `levenberg-marquardt`, serratus anterior (middle part).



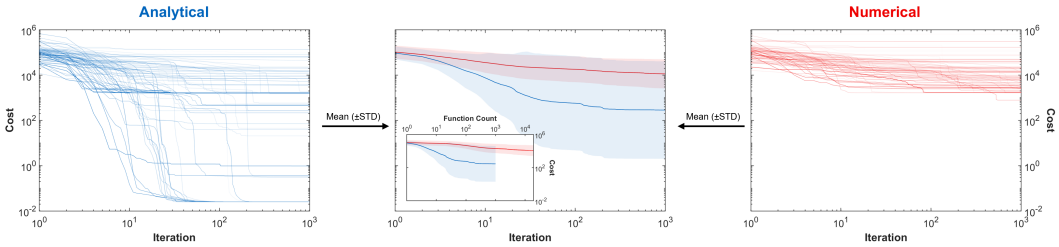
Appendix 3—figure 3. `levenberg-marquardt`, latissimus dorsi (thoracic part).



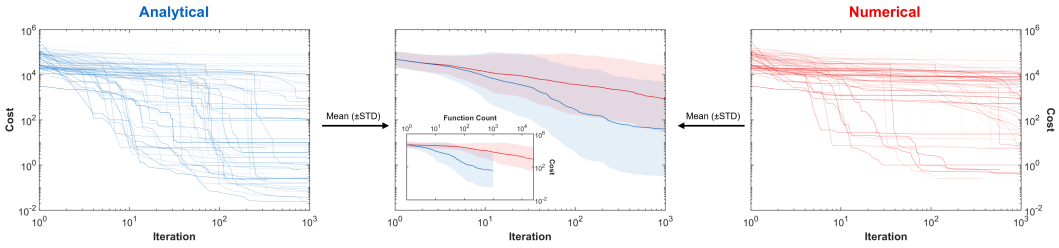
Appendix 3—figure 4. `trust-region-reflective`, serratus anterior (superior part).



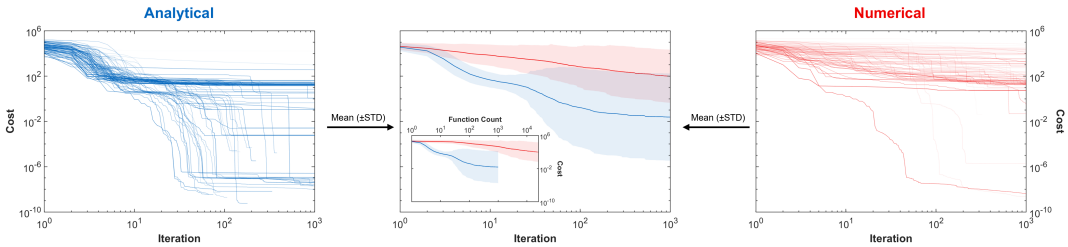
Appendix 3—figure 5. `trust-region-reflective`, serratus anterior (middle part).



Appendix 3—figure 6. `trust-region-reflective`, latissimus dorsi (thoracic part).



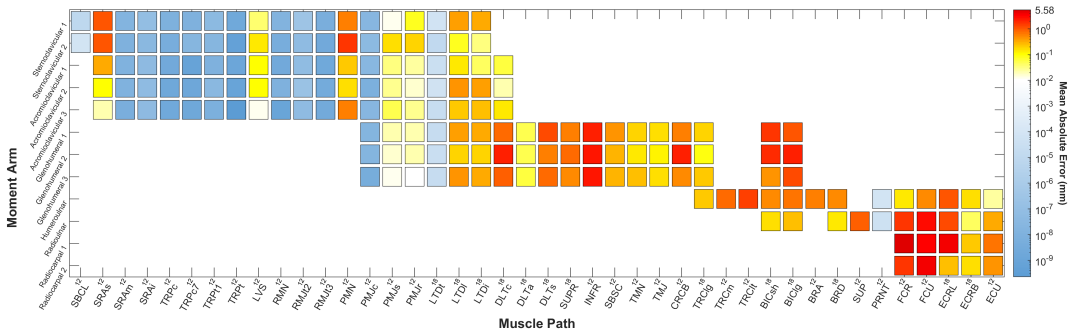
Appendix 3—figure 7. `trust-region-reflective`, extensor carpi radialis brevis.



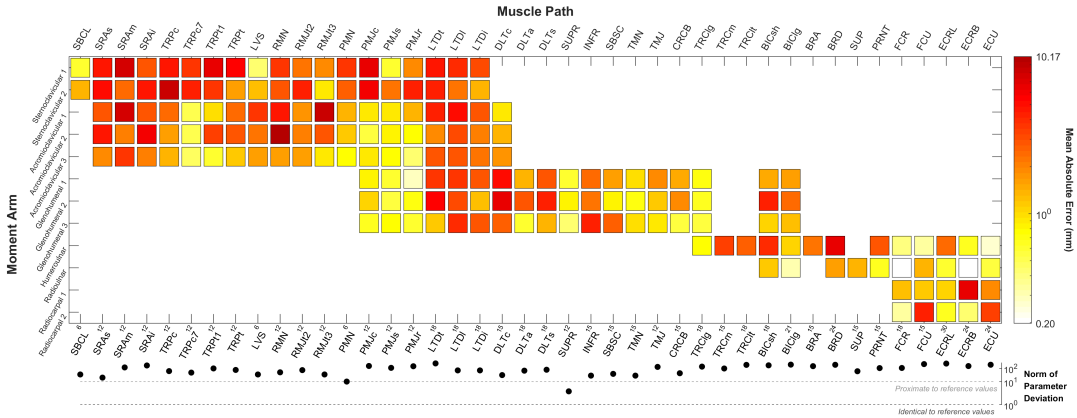
Appendix 3—figure 8. `trust-region-reflective`, extensor carpi ulnaris.

Appendix 4

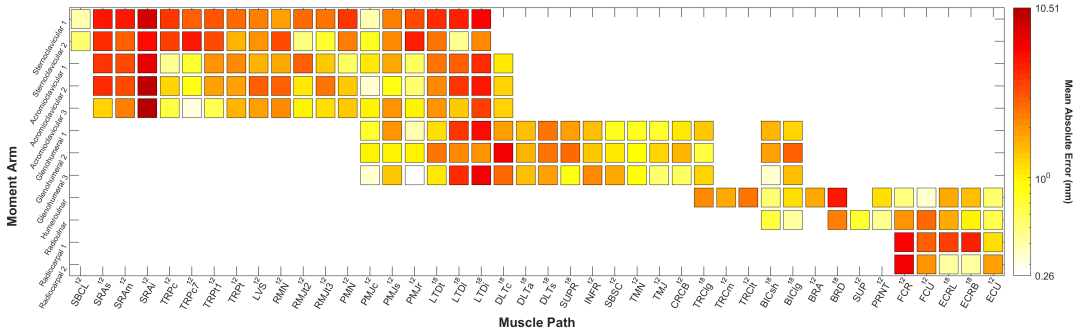
The following figures are supplements to ***Figure 5***, showing the validation errors in 182 effective moment arms of 42 muscle paths. The magnitude of the error is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle path. The norm of parameter deviation is not computable for the modified parameter structure. See ***Appendix 2*** for nomenclatures in the axes.



Appendix 4—figure 1. Modified parameter structure, noise-free calibration data.



Appendix 4—figure 2. Original parameter structure, noisy calibration data.



Appendix 4—figure 3. Modified parameter structure, noisy calibration data.